

Evaluación del módulo 4

Proyecto: Suite de Automatización Funcional

Situación inicial

Unidad solicitante: Área de Calidad y Testing de una empresa de desarrollo de software.

En esta situación, se requiere validar los flujos críticos de una aplicación web que ofrece registro de usuarios e inicio de sesión. El objetivo principal es garantizar que la plataforma cumpla los requerimientos funcionales mínimos antes de su liberación a producción.

El equipo ha identificado que, en iteraciones anteriores, varios defectos en la pantalla de registro e inicio de sesión se descubrieron tardíamente. Esto provocó costos elevados de corrección y demoras en la entrega. Como respuesta, la gerencia ha solicitado la implementación de pruebas automatizadas con Selenium para **reducir el tiempo de validación**, mejorar la cobertura de pruebas y **asegurar la calidad** en cada sprint de desarrollo.

Nuestro objetivo

Desarrollar y ejecutar una **suite de automatización funcional** que se enfoque en:

1. La **validación del formulario de registro** (campos obligatorios, reglas de negocio, mensajes de error, etc.).
2. La **validación de inicio de sesión** (login con credenciales válidas e inválidas, y comportamiento ante bloqueos o cambios de contraseña).
3. La **generación de evidencias de ejecución** (capturas de pantalla, logs, reportes de resultados).

Este proyecto busca dejar establecida una **base automatizada** que pueda extenderse a otros módulos y funcionalidades de la aplicación, sirviendo como pilar de las pruebas de regresión en ciclos futuros.

Requerimientos

1. Automatización con Selenium

- Usar Selenium WebDriver (Java, Python o el lenguaje que prefieran) con Visual Studio Code u otro IDE gratuito.
- Implementar scripts que contemplen la manipulación de elementos web (inputs de texto, botones, alerts, etc.) siguiendo las buenas prácticas descritas en los manuales.

2. Escenarios de prueba

- **Registro de usuario:** Validar campos obligatorios, formato de correo, contraseñas seguras y manejo de alertas.
- **Inicio de sesión:** Verificar ingreso con credenciales válidas, credenciales inválidas, y comportamiento ante varias fallas sucesivas (e.g., bloqueo de cuenta).

3. Cross-Browser

- Preparar la prueba para que se ejecute al menos en Chrome y Firefox (pueden emplear WebDriverManager para la gestión automática de drivers).

4. Datos de prueba

- Utilizar un **mecanismo de incorporación de datos** (por ejemplo, DataProvider en TestNG o parámetros externos en JUnit) para probar múltiples usuarios/contraseñas.
- Se recomienda el uso de **archivos CSV o Excel**, o en su defecto, un mock de base de datos con SQLOnline para consultas simples (opcional).

5. Reportes y evidencia

- Incluir capturas de pantalla y logs en caso de error.
- Producir un informe final (puede ser un archivo HTML o PDF) que consolide los resultados.

Entregables ✓

1. Código fuente

- Proyecto con la estructura adecuada (por ejemplo, un repositorio de GitHub) que incluya:
 - Scripts de Selenium WebDriver (clases, métodos de prueba, configuración).
 - Datos de prueba (CSV, Excel o tablas simuladas en SQLOnline).
 - Configuración para cross-browser testing (Chrome y Firefox como mínimo).

2. Documentación técnica

- Breve descripción de la arquitectura de pruebas (estructuras de carpetas, patrón POM si se aplica).
- Instrucciones para ejecutar los tests localmente (incluyendo instalación de dependencias y drivers, si no se usa WebDriverManager).

3. Evidencias de ejecución

- Capturas de pantalla o logs que muestren la ejecución exitosa y los resultados de los distintos escenarios (registro válido, registro inválido, login válido/invalidado).
- Reporte final (HTML/PDF) con el resumen de las pruebas y los pasos críticos verificados.

4. Presentación final

- Pequeña explicación (puede ser un video corto o un README detallado) donde se demuestre el funcionamiento de la suite de automatización y se destaquen las lecciones aprendidas.

5. Link entregable en Moodle o repositorio

- Subir el proyecto a un repositorio (GitHub/GitLab/Bitbucket) que contenga todos los componentes descritos.

Tras completar este proyecto, se recomienda a los participantes **agregarlo a su portafolio profesional**. Para ello:

- Incluyan en su repositorio una **descripción clara** del objetivo y la importancia de las pruebas automatizadas en proyectos reales.
- Destaquen las **técnicas de POM** y **mecanismos de datos de prueba** (DataProviders, CSV/Excel) implementados, junto a la **ejecución en múltiples navegadores**.
- Sumen capturas de pantalla o GIFs de la ejecución para mostrar el flujo de prueba de registro e inicio de sesión de forma concisa y atractiva.

Este proyecto evidencia conocimientos clave sobre **automatización funcional con Selenium**, desde la configuración inicial en un IDE gratuito hasta la orquestación de pruebas en varios navegadores, aportando un gran valor en procesos de reclutamiento y oportunidades laborales en QA o desarrollo de software.